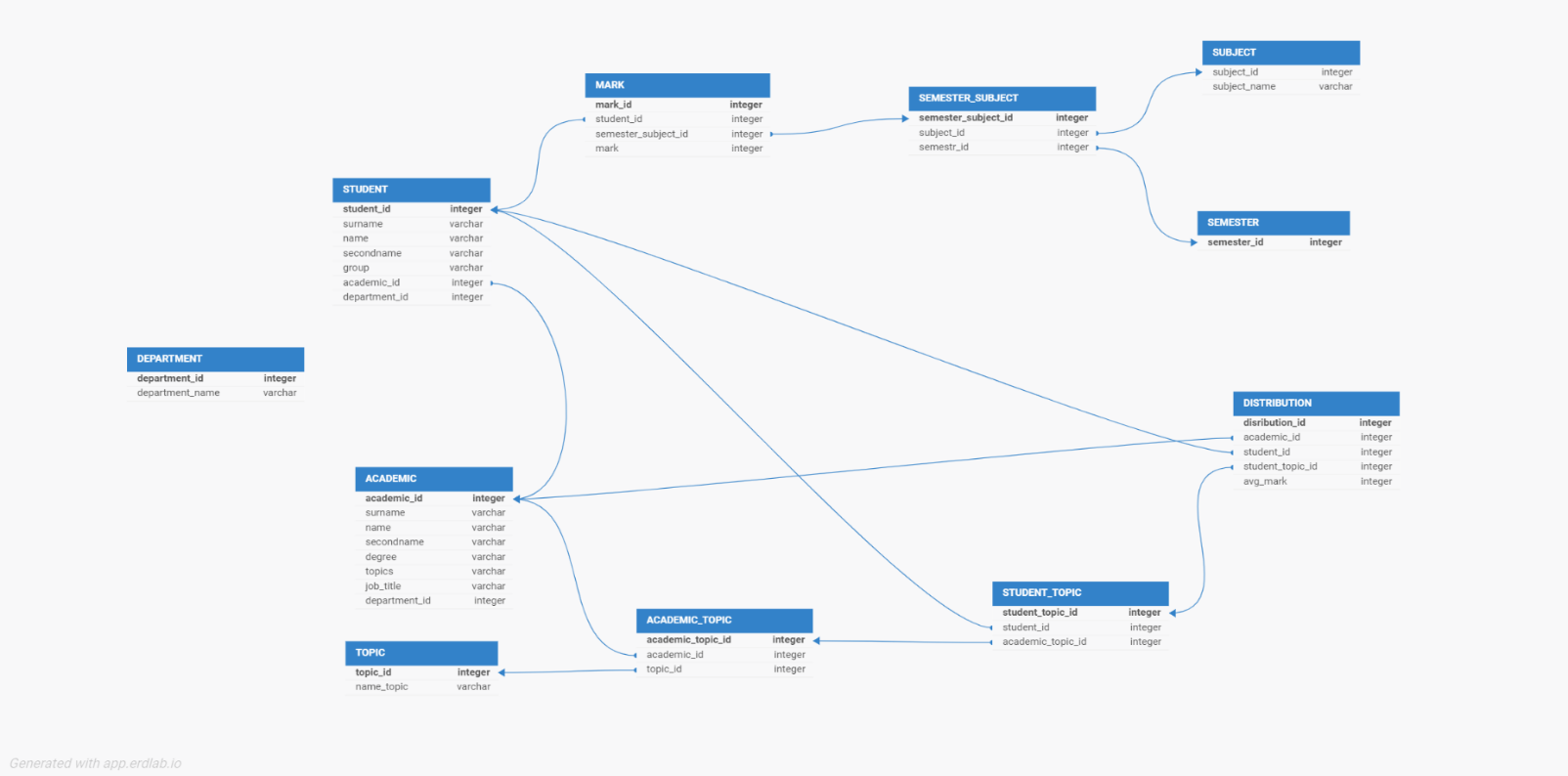
**Информационное моделирование:**



Вы начинаете с рисования схемы БД, но ER-диаграмма описывает не БД, а сущности из задачи и их взаимосвязи. ER-диаграмма позволяет уяснить, какими сущностями и их атрибутами мы оперируем в задаче, как и через что они между собой связаны. Схема БД просто графически показывает, как эти сущности и связи представлены в хранилище.

Сущности:

Тема (Topic): представляет собой список всех тем ВКР, которые существуют.

   - Атрибуты:

     - topic\_id (первичный ключ): уникальный идентификатор темы.

     - name\_topic: название темы.

Есть конкретные темы, а есть тематики, то есть области, из которых формируются темы. Например, «Информационные системы» — тематика, а «ИС для распределения тем ВКР» — тема. По смыслу сущность «Topic» есть тематика.

Предмет (Subject): представляет собой отдельный предмет, который может изучаться в одном или нескольких семестрах.

   - Атрибуты:

     - subject\_id: уникальный идентификатор предмета.

     - subject\_name: название предмета.

К этой и связанным с предметами таблицам: зачем такое сложное представление нужно в решаемой задаче? И наоборот, все ли учтено? Если есть руководитель, который занимается машинным обучением, и у него осталось одно свободное место, на него логично рекомендовать студента, у которого выше баллы по предметам, связанным со статистикой. Сможет ли алгоритм учесть это? Нет, не сможет, потому что для тематик не предусмотрено атрибутов, которые показывали бы, оценки по каким предметам важны. С другой стороны, зачем требуется таблица семестров, если это просто номер семестра?

Студент (Student): представляет собой отдельного студента, который может изучать различные предметы и выбирать темы ВКР

   - Атрибуты:

     - student\_id (первичный ключ): уникальный идентификатор студента.

     - surname: фамилия студента.

     - name: имя студента.

     - secondname: отчество студента.

«Имя», «фамилия» и «отчество» по-английски будут «first name», «last name» и «patronymic».

     - group: группа, в которой обучается студент.

     - academic\_id: идентификатор академического руководителя студента.

     - department\_id: идентификатор кафедры, на которой обучается студент.

Кафедра (Department): представляет собой кафедру, на которой обучаются студенты и к которой принадлежат преподаватели.

    - Атрибуты:

    - department\_id (первичный ключ): уникальный идентификатор кафедры.

    - department\_name: название кафедры.

Сущность «Кафедра» можно убрать на первом этапе. При необходимости элементарно добавляется, а сейчас только создаст лишнюю работу.

Академический руководитель (Academic): представляет собой академического руководителя, который ведет одну или несколько тем.

Название не очень, потому что «академический руководитель» не говорят, говорят просто «руководитель» или «научный руководитель» (у магистров и выше). По-английски это называется «adviser».

   - Атрибуты:

     - academic\_id (первичный ключ): уникальный идентификатор академического руководителя.

     - surname: фамилия академического руководителя.

     - name: имя академического руководителя.

     - secondname: отчество академического руководителя.

     - degree: степень академического руководителя.

     - topics: темы, которые ведет академический руководитель.

Такой атрибут может быть в ER-модели, но вы описываете БД, а там такого нет, там есть связь с топиками через промежуточную таблицу.

     - job\_title: должность академического руководителя.

Зачем нужна для решаемой задачи?

     - department\_id: идентификатор кафедры, на которой работает академический руководитель.

Академические темы (AcademicTopic): представляют собой темы ВКР, которые может вести академический руководитель.

   - Атрибуты:

     - academic\_topic\_id (первичный ключ): уникальный идентификатор академической темы, которые может вести академический руководитель.

     - academic\_id: идентификатор академического руководителя, который ведет тему.

     - topic\_id: идентификатор темы, которую ведет академический руководитель.

Оценки (Mark): представляет собой оценку, которую студент получил за выполнение задания или экзамена.

   - Атрибуты:

     - mark\_id (первичный ключ): уникальный идентификатор оценки.

     - student\_id: идентификатор студента, который получил оценку.

     - semester\_subject\_id: идентификатор предмета и семестра, за который студент получил оценку.

     - mark: оценка, которую студент получил.

Семестр (Semester): представляет собой семестр, в течение которого студенты изучают определенные темы и предметы.

   - Атрибуты:

     - semester\_id (первичный ключ): уникальный идентификатор семестра.

Семестр-предмет (SemesterSubject): представляет собой предмет, который изучается в определённом семестре.

   - Атрибуты:

     - semester\_subject\_id (первичный ключ): уникальный идентификатор семестра-предмета.

     - subject\_id: идентификатор предмета, который изучается в семестре.

     - semestr\_id: идентификатор семестра, в течение которого изучается предмет.

Темы студента (StudentTopic): представляет собой темы, которые выбрал студент при анкетировании.

    - Атрибуты:

    - student\_topic\_id (первичный ключ): уникальный идентификатор темы студента.

    - student\_id: идентификатор студента, который выбрал темы.

    - academic\_topic\_id: идентификатор академической темы, которую выбрал студент из тем всех преподавателей кафедры.

Здесь и в других таблицах связей: вы уверены, что нужны суррогатные первичные ключи самих связей?

Распределение (Distribution): представляет собой распределение студентов по темам и предметам.

   - Атрибуты:

     - disribution\_id (первичный ключ): уникальный идентификатор распределения.

     - academic\_id: идентификатор академического руководителя, который ведет тему ВКР.

     - student\_id: идентификатор студента, который выбрал тему ВКР.

     - student\_topic\_id: идентификатор темы ВКР, которую выбрал студент.

     - avg\_mark: средняя оценка студента по предметам за несколько семестров.

Зачем нужна для решаемой задачи?

**Связи между сущностями**:

Студент может выбрать несколько тем (STUDENT\_TOPIC).

Предмет может быть связан с одним или несколькими семестрами (SEMESTER\_SUBJECT).

Семестр может быть связан с одним или несколькими предметами (SEMESTER\_SUBJECT).

Академический руководитель может вести одну или несколько тем (ACADEMIC\_TOPIC).

Тема может быть связана с одним или несколькими академическими руководителями (ACADEMIC\_TOPIC).

У Студента есть итоговые оценки за семестры

(MARK).

Кафедра может быть связана с одним или несколькими академическими руководителями и студентами (DEPARTMENT).

**Требования к СУБД:**

Требования к чему бы то ни было формируются иначе: «приложению нужно делать X, поэтому база данных должна иметь Y». Вы же отчасти перечислили возможные характеристики СУБД (2, 3, 4, 5, 6), отчасти написали то, что не имеет отношения к задаче (1).

Выбор делается так: составляется таблица, где по одному измерению выписаны требования, по другому — альтернативы (варианты), а в ячейках указано, соответствует или нет (иногда с комментарием). Оба эти элемента характерные для ВКР, их надо будет представить правильно. Вы ничего этого не сделали, а между MySQL и Postgres выбрали по критерию, о котором ничего не говорили ранее (и о том, почему он важен).

1. Многопользовательская система: поддержка аутентификации и авторизации пользователей (преподавателей и студентов) на уровне приложения, а в СУБД хранится информация о пользователях и правах.
2. Реляционная база данных: для хранения информации о преподавателях, тематиках, студентах и их успеваемости.
3. Поддержка связей между сущностями: поддержка связей между сущностями (преподаватель-тематика, студент-тематика, студент-успеваемость) с помощью внешних ключей и контроля ссылочной целостности.
4. Конкурентность доступа: поддержка одновременного доступа нескольких пользователей к базе данных.
5. Конкурентность запросов: поддержка одновременной обработки нескольких запросов к базе данных без конфликтов.
6. Транзакции: поддержка транзакций для обеспечения целостности данных и предотвращения конфликтов при одновременной обработке запросов.
7. Контроль ссылочной целостности: поддержка контроля ссылочной целостности для обеспечения целостности данных и предотвращения записи несуществующих ссылок.

Исходя из вышеперечисленных требований К СУБД, я выбрал MySQL так как:  
MySQL более легкая в использовании, чем PostgreSQL, а SQLite имеет ограничения на количество одновременных подключений, ограничения на количество одновременных запросов.

Если вы уже погрузились в вопросы выбора СУБД, то напоминаю, что MySQL под Oracle, а свободный форк называется MariaDB. Так как задача в целом решается для российских вузов, актуален вопрос, что СУБД должна быть свободная либо отечественная.